REMARKS

In response to the Office Action of April 5, 2006, the Applicants submit this Reply. In view of the foregoing amendments and following remarks, reconsideration is requested.

Claims 1-6 remain in this application, all of which claims are independent. No fee is due for claims for this amendment.

Summary of Invention

The present invention provides a single context-based DMA engine connected to the memory system that implements the logic for each DMA function only once, and switches parameter sets as needed to service various DMA requests from different channels. (p1,ll. 18-21) Arbitration is performed at the DMA request level. After a DMA channel is selected for service, the parameters for that channel's transfer are retrieved from a central context block, the data transfer is queued to the memory system, and the parameters are updated and stored back to the central context block. The DMA system also may have a buffer control unit (BCU) that permits DMA channels to be linked together in a flow controlled system to reduce latency. The buffer control unit allows independent flow control between write and read DMA channels accessing the same data, preventing underflow or overflow of data during simultaneous DMA operations. (page 1, ll. 28-30). The BCU resource tracks the amount of data in the buffer and other buffer state information, and flow-controls the DMA engine(s) appropriately based on parameters that are set up within the BCU resource. Multiple read or write DMA channels also may be linked by the same BCU, so that, for example, two DMA channels could write into one buffer, which in turn is read out by one DMA channel that uses the data from both the input channels. To control data flow, neither the sender nor the receiver requires any knowledge of each other. The sender and receiver each use knowledge of the BCU resource associated with the buffer being used by the given DMA channel. (page 2, lines 1-8). For any given combination of operations to be performed, the data transfers to be performed to support those operations are determined by the application program executing on the CPU. The CPU then allocates the appropriate buffers, and programs the DMA contexts for each channel and BCUs for each buffer. After setting up the DMA operations and the buffers, the data can be processed. Once the CPU initiates the data flow, no further intervention is required by the application or CPU in order for the data to be routed to its destination, and processed through the intermediate steps. Moreover, the DMA and

BCU controllers impose an autonomous flow control mechanism that ensures that data is sequenced properly through the processing steps without further attention from the application program, and with a minimum of latency-based delays. (page 4, lines 12-21).

Rejections Under 35 U.S.C. §102

Claims 1-6, of which claims 1 and 4 are independent, were rejected under 35 U.S.C. §102(b) as being anticipated by Kabenjian (U.S. Patent 5,613,162.) The rejection is respectfully traversed.

In particular, the Action asserts that Kabenjian (at col. 8, lines 45-52), teaches *a plurality of ports, wherein each port has an associated buffer for temporarily storing data transferred through the port, and wherein each port has an associated direct memory access channel.* Applicants respectfully suggest that this is a misreading of that passage of Kabenjian, which discloses "four independent DMA channels" (emphasis added.) Kabenjian discloses a buffering architecture that includes buffers between the I/O devices and the buffer and between the buffer and the memory in order to permit "transfers between the two buffers to occur in bursts to optimize the transfer and to reduce the amount of time that the bus is needed for the transfer." (Abstract). Kabenjian's system makes the I/O bus "available for transfers by the other DMA channel and by other devices on the bus." (Abstract). With reference to FIG. 1, Kabenjian discloses an architecture for independent data transfers on a single, shared port, "primary I/O interface 150", and it is the optimal use of I/O bus 152 with which Kabenjian appears primarily concerned. Primary interface 150 represents a conventional I/O interface, rather than a plurality of ports, as that term is understood in the art. Nor do Kabenjian's DMA register blocks (300,320,340,360) comprise a *plurality of ports* (such as ports 110a-110d in Applicants' Figure 1), each having *an associated buffer* (such as FIFOs 112a-112d) *for temporarily storing data transferred through the port.* Kabenjian's register blocks comprise a part of his DMA controller 202 ("the DMA controller comprises a first set of DMA control registers which controls transfer of data...", col. 3, ll. 26-28), whereas in claim 1 the *direct memory access controller* is clearly a separate limitation representing functionally distinct elements from *the plurality of ports*.

The distinctions above inter-relate with a second major distinction between the disclosure of Kabenjian and the invention as recited in claim 1, namely the operation of the DMA controller. Claim 1 further recites *wherein the direct memory access controller stores parameters*

defining the direct memory access operations for each port, and wherein after a request is received from a port the direct memory access controller loads the parameters for the current direct memory access operation for the port to enable the port to access the memory. The Action asserts that such operation is disclosed, starting at col. 8, line 66 and continuing through col. 10, line 10. As noted above, the Applicants' DMA controller imposes an autonomous flow control mechanism, following data transfer initiation, that ensures that data is sequenced properly through the processing steps without requiring further attention from the CPU, by the *direct memory access controller's loading of the context parameters in response to requests*. In contrast to the claimed invention, it is clear from the passage cited in the Action and portions thereafter that it is the integral involvement of Kabenjian's CPU 100 and memory interface 204 that enable a data transfer from memory unit 120 to I/O device IDS controller 154, using the DMA unit 200, rather than the *direct memory access controller* operating autonomously after having its logic set up by the CPU.

According to Kabenjian, during initiation, the CPU 100: writes a destination address, size of the transfer and other control data to the DMA controller 202; requests control of the memory bus 110; executes one or more read/write cycles addressed to the IDE device 156; initializes the DMA unit 200; writes to source, destination and count registers A; and writes go and direction data to the command register. (col. 10, ll. 5-61). After setting up the register set A 300 for a first DMA transfer, the CPU 100 either resets the first DMA channel for a second DMA transfer (col. 10, ll. 62-64), or uses a two-deep shadowing technique of "writing appropriate values to register set B of the first register block while the DMA unit is performing the previously initialized DMA transfer" (col. 11, ll. 3-6) or uses multiplexing to quickly switch between register set A or B. (col. 11, ll. 19-21). In contrast, the DMA of the present invention *loads the stored parameters* (addressing, counters, pointers to buffer control units, etc.) that enable the data access between the port and the memory, without requiring the assistance of the CPU. In addition, Kabenjian's architecture appears to impose a critical timing requirement on the CPU's writing of transfer parameter data to the register sets that is reduced or eliminated in the presently claimed invention, which employs linked lists (CRB 118 and BCU 120) of parameter information easily accessible by the DMA controller.

In light of Kabenjian's failure to teach or suggest all of the limitations of claim 1 as described above, Applicants respectfully submit that claim 1 is patentable over Kabenjian.

With regard to claim 2, Kabenjian also fails to disclose a central parameter store for storing parameters for each of a plurality of DMA channels corresponding to each of the plurality of ports. As noted above, Kabenjian does not disclose a plurality of ports, and the parameters that are stored in register sets A and B of register blocks 300-360 may be frequently over-written by the CPU during a data transfer operation. (col. 9, ll. 4-48). Whereas, in the presently claimed invention, the parameter information for each of a plurality of DMA channels corresponding to each of the ports is stored during initial set-up by an application program, and thereafter any request for data transfer to or from a port selected for servicing by the DMA controller results in a loading of the parameters from the central parameter store. Certain parameters associated with a channel and port may be updated as a result of buffer state, etc., but otherwise can be viewed in a sense as dedicated to data transfers to/from that channel and port, which increases operational efficiency.

With regard to claim 3, the passage of Kabenjian cited in the Action fails to disclose a *DMA controller comprised of means for queuing a memory operation*, *means for updating the stored parameters,* and *means for fetching and storing parameters in the central parameter store*. As noted above, Kabenjian discloses programmable register blocks that may employ internal "shadowing" (register-to-register quick copying), but that is not means for *queuing* a memory operation using the term "queuing" as understood by those of skill in the art. As also noted above, Kabenjian's architecture involves regular CPU involvement in writing parameter information to the register blocks, in contrast to the claimed invention, wherein the direct memory access controller <u>itself</u> comprises means for updating, fetching and storing the parameters for the plurality of channels associated with the plurality of ports. In light of the foregoing, and in light of the dependency of claims 2 and 3 from patentable claim 1, Applicants respectfully submit claims 2 and 3 are similarly patentable over the cited art.

With respect to independent claim 4, the instant Action asserts that Kabenjian discloses a DMA controller that *receives information from a DMA context memory specifying parameters for writing data from the first device to the memory and for reading data from the memory to the second device*. Claim 4 has been amended to clarify a novel aspect of parallelism present in the invention that is not disclosed by Kabenjian, more specifically, *wherein the DMA controller is adapted to enable simultaneous reading and writing memory access operations*. As noted above

and in Applicants' specification, "The DMA system also may have a buffer control unit (BCU) that permits DMA channels to be linked together in a flow controlled system to reduce latency. The buffer control unit allows independent flow control between write and read DMA channels accessing the same data, preventing underflow or overflow of data during simultaneous DMA operations." (page 2, ll. 28-32). DMA operations to or from a buffer in memory may be monitored, and flow-control set up appropriately for multiple read or write DMA channels to access the same buffer. The cited passages of Kabenjian do not disclose such operation, and it is therefore submitted that claim 4, as amended, is patentable over the cited art.

With regard to claims 5 and 6, it is noted that Kabenjian's FIFO control unit 606 "indicates whether the write FIFO 602 or the read FIFO 604 is empty or full." (col. 9, ll. 47-49). This is both structurally and functionally distinct from *a buffer control unit for communicating to the DMA controller an indication of an amount of data <u>written into the memory</u> by the first device through the DMA controller and for communicating to the DMA controller an indication of the amount of data <u>read from the memory</u> by the second device through the DMA controller*. While Kabenjian's control unit monitors the FIFOs (602, 604) of an external I/O device, the *buffer control unit* is monitoring one or more buffers *of the shared access memory* (e.g., memory 100 in Figure 1 of Applicants' application.) Applicants' buffer control unit maintains state information about the buffers allocated in the memory to be accessed by the distinct devices, enabling simultaneous DMA operations to be performed on the same memory buffer(s). Claims 5 and 6 also depend from claim 4, as amended. In light of the foregoing, Applicants request reconsideration and withdrawal of this ground for rejection of claims 5 and 6.

CONCLUSION

In view of the foregoing amendments and remarks, this application should now be in condition for allowance. A notice to this effect is respectfully requested. If the Examiner believes, after this reply, that the application is not in condition for allowance, the Examiner is requested to call the Applicants' attorney at the telephone number listed below.

If this response is not considered timely filed and if a request for an extension of time is otherwise absent, Applicants hereby request any necessary extension of time. If there is a fee

occasioned by this response, including an extension fee, please charge any fee to **Deposit Account No. 50-0876.**

Respectfully submitted,

Avid Technology, Inc.

By John A. Hamilton

John A. Hamilton
Registration No. 48,946
Avid Technology, Inc.
One Park West
Tewksbury, MA 01876
Tel.: (978) 640-6789